목차

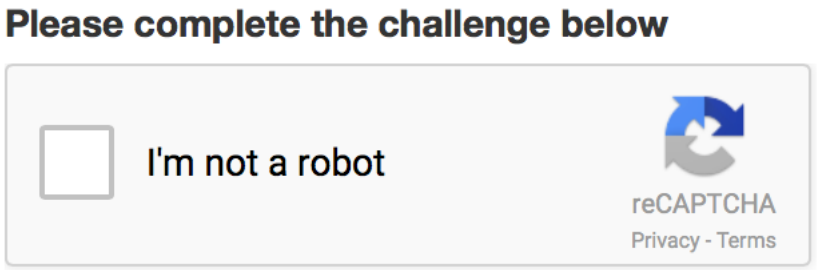
table of contents

1

주제 선정

CAPTCHA vs 자동화 매크로

가장 강한 방패 vs 가장 강한 창의 대결



주제 선정

What is CAPTCHA?

Completely Automated Public Turing test to tell Computers and Humans Apart

이미지 혹은 오디오를 주고 정답을 맞히게 함으로써
사람과 컴퓨터를 판별

매크로를 개발 및 사용하고자 하는 입장에서는
파훼해야 하는 대상,
악성적인 공격과 매크로 이용을 막아야 하는
입장에서는 절대 **파훼되어서는 안 되는 보안장치**

튜링 테스트
질의자가 인간과 컴퓨터를 대상으로 정해진 시간 안에 대화를
나누는 방식으로 이루어지는데, 대화를 통해 인간과 컴퓨터를
구별해내지 못하거나 컴퓨터를 인간으로 간주하게 된다면
해당 기계는 인간처럼 사고할 수 있는 것으로 보는 인공지능 판별법

2

분석 방법 설정

Part 2

분석 방법 설정

1. 각종 데이터를 이용하여
CAPTCHA를 해석할 수 있는 모델 훈련

2. 실제 CAPTCHA 이미지로 예측 & 성능평가

3. 다양한 개선 방안과 방법론을 적용해 보며
정확도 개선 시도

OCR model for reading Captchas

Author: A_K_Nain
Date created: 2020/06/14
Last modified: 2020/06/26
Description: How to implement an OCR model using CNNs, RNNs and CTC loss.

ⓘ This example uses Keras 3

View in Colab • **GitHub source**

Introduction

This example demonstrates a simple OCR model built with the Functional API. Apart from combining CNN and RNN, it also illustrates how you can instantiate a new layer and use it as an "Endpoint layer" for implementing CTC loss. For a detailed guide to layer subclassing, please check out this page in the developer guides.

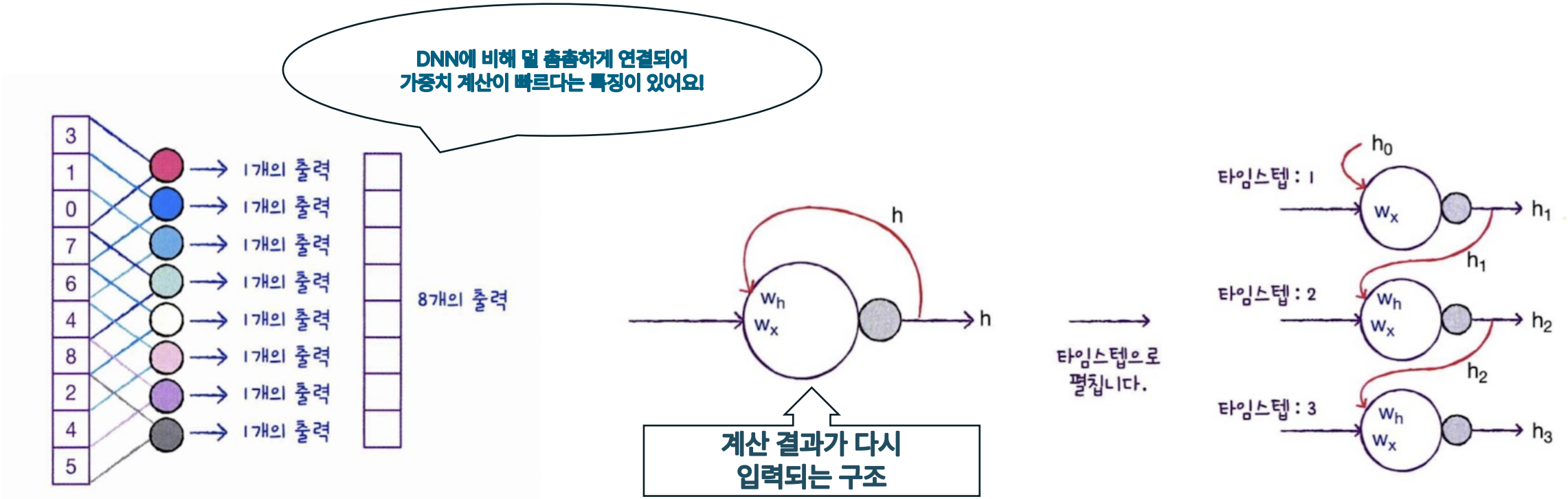
Keras에서 제공하는 OCR 분석 방식인 CRNN 모델을 활용하기로 결정

3

CRNN 모델이란?

Part 3

CRNN 모델이란?



Convolution Layer를 사용하여
이미지의 특징을 학습하는 CNN

순환적인 구조를 활용하여
순차적인 패턴을 학습하는 RNN

→ 두 방식의 특징을 활용하여 이미지 내의 텍스트 인식에 활용할 수 있음

4

1차 시도: 범용적 모델

Part 4

범용적인 Captcha 만들어보기

PARSA SAMADNEJAD · UPDATED 3 YEARS AGO

CAPTCHA Dataset

+113,000 alphanumeric colorful images

Data Card | Code (11) | Discussion (0)

About Dataset

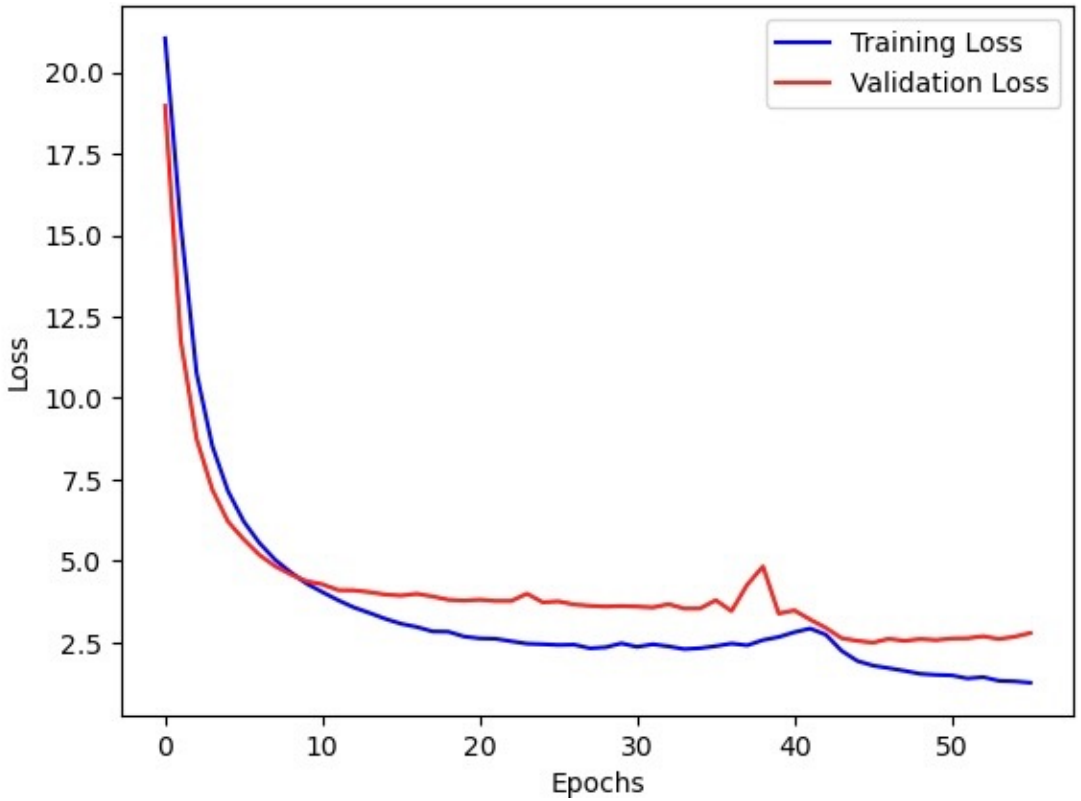
Content

Captcha Dataset contains more than 113,000 colorful 5-character images.

- Kaggle에 있는 약 113,000개의 5글자 CAPTCHA를 Dataset으로 사용
- Keras의 코드와 다르게 Batch에 대한 정규화를 진행
  → 과적합 방지, 학습 속도 개선 목표

Part 4

범용적인 Captcha 만들어보기



- Batch size = 32 (with normalization)
- Learning Rate = 0.001
- Validation loss = 1.9282

- 대소문자의 형태 차이가 크기에 불과한 경우, 예측에 실패하는 경우 존재(YCAR2, R1zfq)
- 애초에 레이블링이 잘못된 경우도 존재함(R1zfq, czjcy)

사이즈도 크고... 꽤나 다양한 데이터가 모인 것 같은데

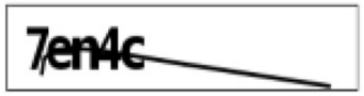
다른 사이트의 CAPTCHA에서도 뚫릴 수 있을까?

Part 4

범용적인 Captcha 만들기

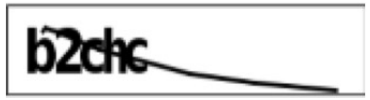
- 위메프

Prediction : [UNK][UNK][UNK][UNK][UNK]
Actual : 7en4c

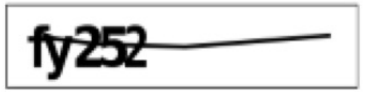


Prediction : c[UNK]4[UNK]b
Actual : 8b3g7

Prediction : [UNK]3[UNK][UNK][UNK]
Actual : b2chc

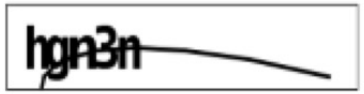


Prediction : 33[UNK][UNK][UNK]
Actual : fy252

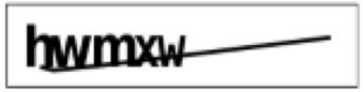


Prediction : [UNK]b[UNK][UNK][UNK]
Actual : gkpg7

Prediction : [UNK][UNK][UNK][UNK][UNK]
Actual : hgn3n

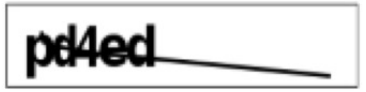


Prediction : [UNK][UNK][UNK][UNK][UNK]
Actual : hwmxw



Prediction : [UNK][UNK][UNK][UNK][UNK]
Actual : knm8d

Prediction : [UNK][UNK][UNK][UNK][UNK]
Actual : pd4ed



Prediction : [UNK][UNK][UNK][UNK][UNK]
Actual : rbdc2

- 스타벅스

Prediction : n[UNK][UNK]2[UNK]
Actual : 5crrh

Prediction : a4[UNK]b[UNK]
Actual : 73n8g

Prediction : b[UNK][UNK][UNK][UNK]
Actual : 8gyef

Prediction : [UNK][UNK][UNK][UNK][UNK]
Actual : affnr

Prediction : [UNK][UNK]55[UNK]
Actual : pb4x8

Prediction : [UNK][UNK][UNK]a[UNK]
Actual : wnhc7

Prediction : [UNK][UNK][UNK][UNK]a
Actual : xhgm7

Prediction : [UNK][UNK]b[UNK][UNK]
Actual : xr8rp

Prediction : [UNK][UNK][UNK]54
Actual : xwc43

Prediction : [UNK][UNK]43[UNK]
Actual : yp32g

사이즈가 커도 뚫을 수가 없다...

한 사이트에 최적화된 CAPTCHA를 만들어보면 어떨까?

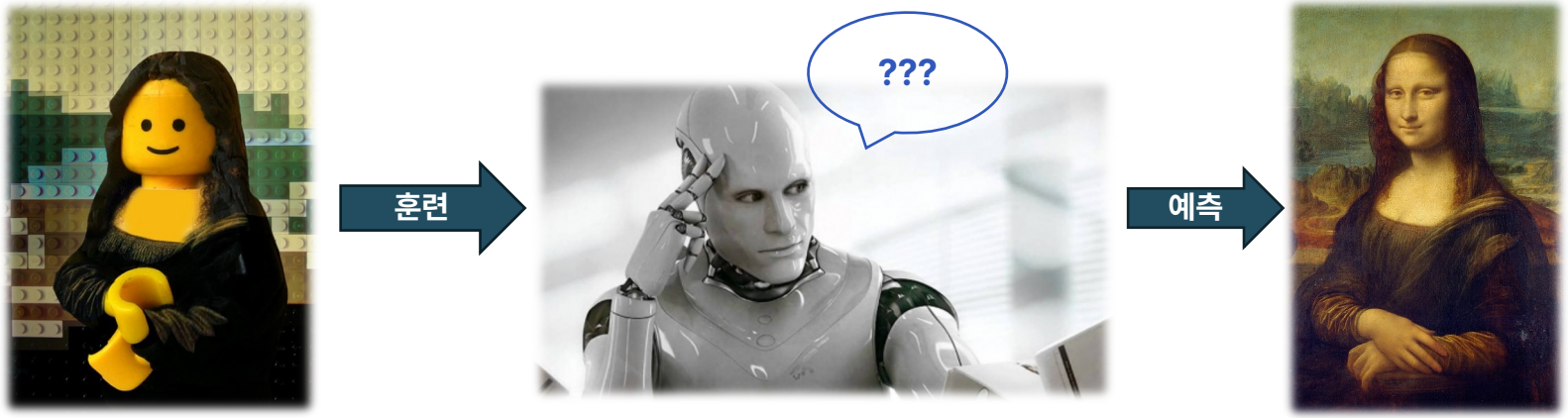
5

2차 시도: 특화 모델

한 CAPTCHA에 특화된 모델을 만들어보고 싶은데...

이미지를 일일이 다운받아 레이블을 달아서 훈련시키기는 번거롭다(~~귀찮다~~)...

이미지를 똑같이 따라서 만들면...
비슷한 효과를 낼 수 있지 않을까?



Part 5

특화 Captcha 만들기

1차 시도 : DCInside Captcha 특화 모델

위 : 원본 이미지(예측 대상)
아래 : 모방 이미지(학습 데이터)

→

```
Epoch 12/100
141/141 [==============================] - 15s 104ms/step - loss: 7.1589 - val_loss: 7.1883
Epoch 13/100
...
Epoch 99/100
141/141 [==============================] - 15s 106ms/step - loss: 7.1424 - val_loss: 7.2043
Epoch 100/100
141/141 [==============================] - 15s 108ms/step - loss: 7.1402 - val_loss: 7.2023
```

노이즈와 색상 등의 영향으로 학습 진행 X(7의 지옥에 갇힌 loss값...)

배경색 제거, 글자색 검정으로 통일

→

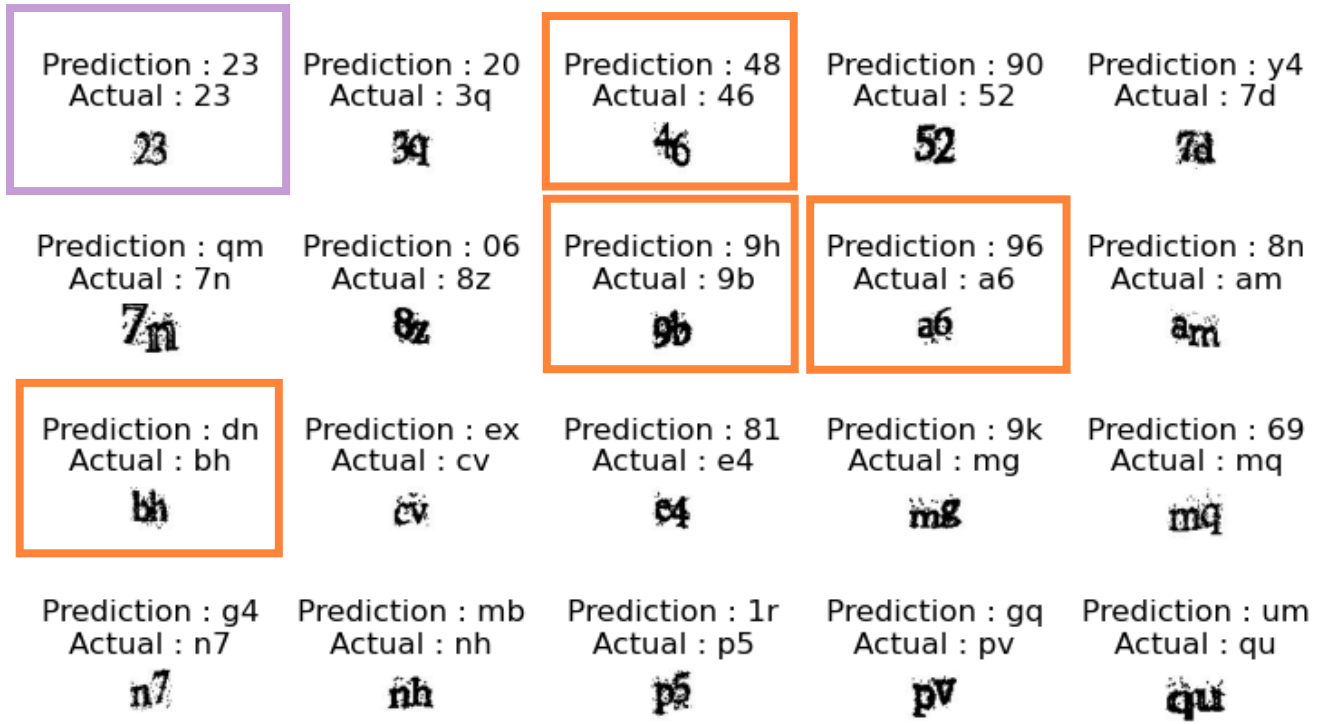
```
loss: 0.1625 - val_loss: 0.2505

loss: 0.1801 - val_loss: 0.1965
```

학습까지는 성공
과연 결과는?

Part 5

특화 Captcha 만들기

예측 결과



보라 : 정답
주황 : 한 글자는 정답

한줄평 : 뭔가 되는 듯 했으나 실패!

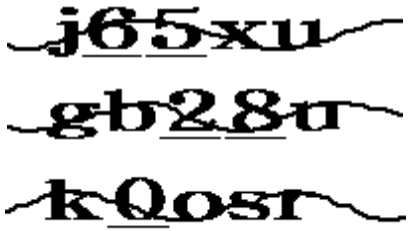
학습 데이터와 실제 이미지 간의 간극이 大
→ 수많은 차이로 인해 예측 모델 정확도 ↓

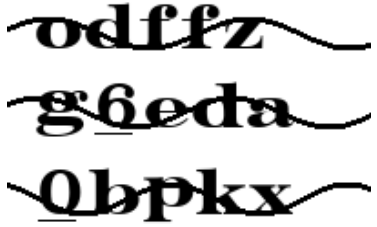
→ 글자 수가 적다고 얕봤다가 큰 코 다쳤다...
조금 더 모방하기 쉬운 캡챠로 Retry!

Part 5

특화 Captcha 만들기

2차 시도 : NETIS 공유기 CAPTCHA 특화 모델

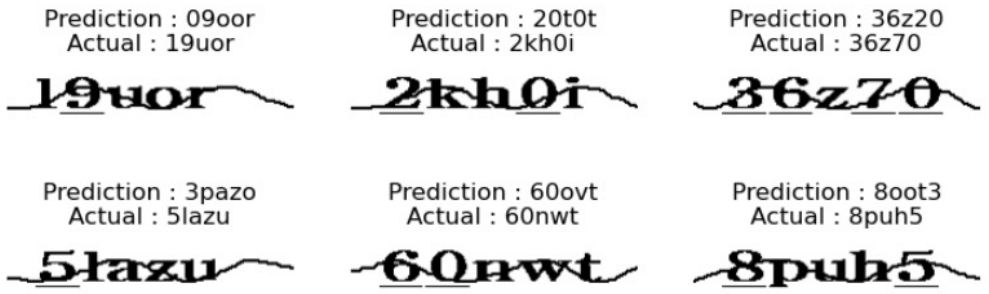




위 : 원본 이미지(예측 대상)
아래 : 모방 이미지(학습 데이터)

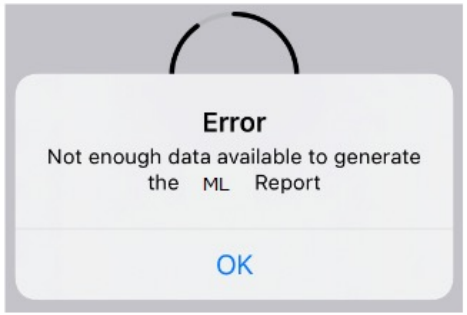
→

모방 데이터 1만개로 학습한 결과



비슷하게는 맞히지만 정답이 없다...

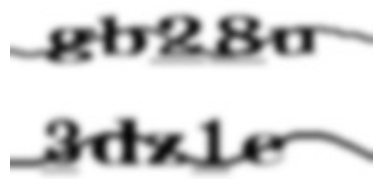
↓



데이터를 3만개로 늘린다면?

+

↓



가우시안 블러를 넣어본다면?

실눈뜨고 보면 똑같은데?

과대적합을 방지하는 장치

Part 5

특화 Captcha 만들기

2차 시도 : NETIS 공유기 Captcha 특화 모델 결과 확인 및 피드백

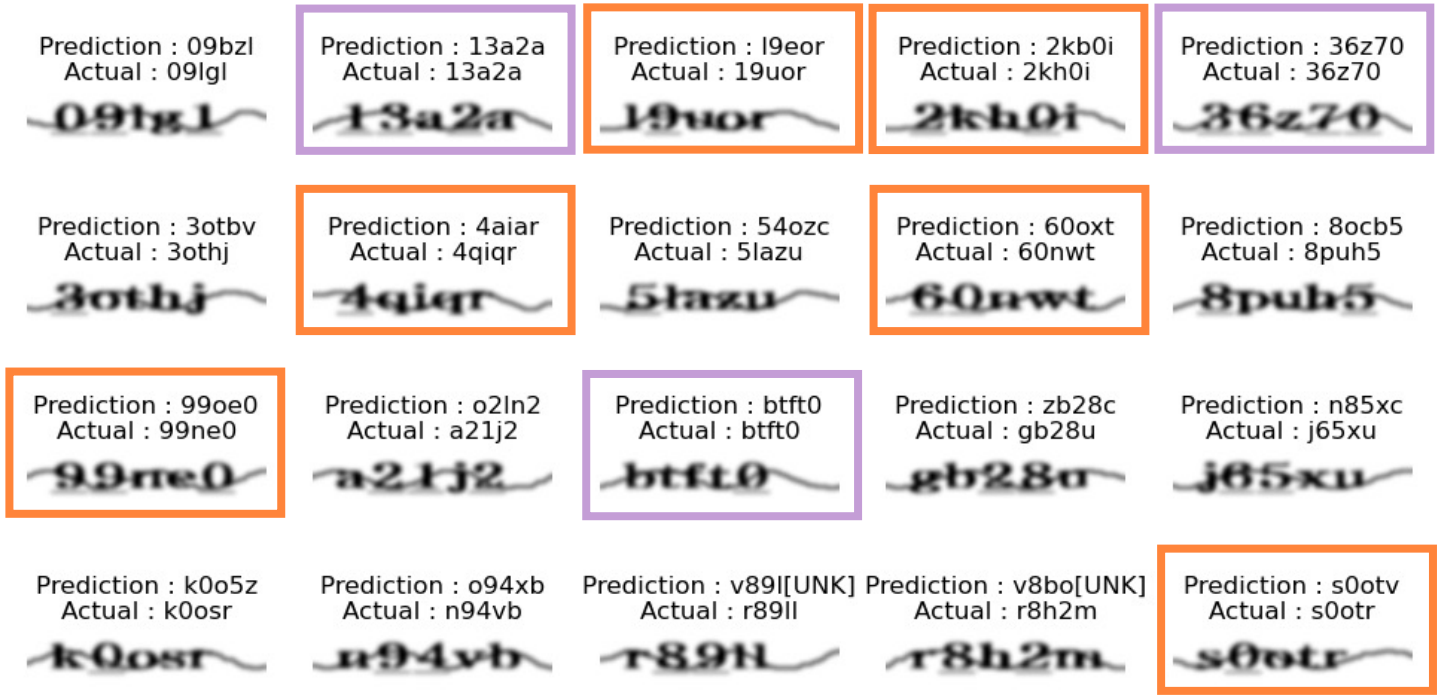
한줄평 : 아쉽긴 하지만 성과는 있다!

데이터를 추가하기만 했을 때 :
2개 정답 **but 인식률 개선 효과 미미**

가우시안 블러까지 추가했을 때 :
3개 정답 + 인식률 대폭 개선

→ 조금 더 **잘 모방한다면** 더 좋은 성과를 얻을 수 있으리라고 확신

예측 결과



보라 : 정답
주황 : 정답에 근접

Part 5

특화 Captcha 만들기

3차 시도 : Melon Ticket 특화 모델 및 매크로 제작

Melon Ticket에서 좌석 예약시 뜨는 실제 화면

구분이 되시나요?

위 : 원본 이미지(예측 대상)
아래 : 모방 이미지(학습 데이터)

1만 개 생성 + 가우시안 블러

Part 5

특화 Captcha 만들기

3차 시도 : Melon Ticket 특화 모델 결과 확인 및 피드백

한줄평 : **대성공!** 놀라울 정도로 잘 맞힌다...

가우시안 블러처리 하지 않았을 때:
몇 개 정답이 있긴했으나 실망스러운 결과

가우시안 블러까지 추가했을 때 :
52개 샘플 중 38개 정답 = **정확도 73%**

→ 정확도를 조금 더 개선하고 **매크로** 제작을 시도

GaussianBlur 적용 유무의 차이

Part 5

특화 Captcha 만들기

Melon Ticket 모델 정확도 개선

1Epoch마다 정확도를 계산

↓

가장 높은 정확도를 기록하는 가중치를 **Save**

Epoch	Accuracy
20	0.923
21	0.577 ?
22	0.942 !
23	0.903
24	0.942 !
25	0.885

가장 좋은 가중치를 적용한 결과

Accuracy : 0.9423076923076923

Incorrect : ['JRLOKF', 'PEXOAF', 'XYMHYI']

6

멜론티켓 매크로

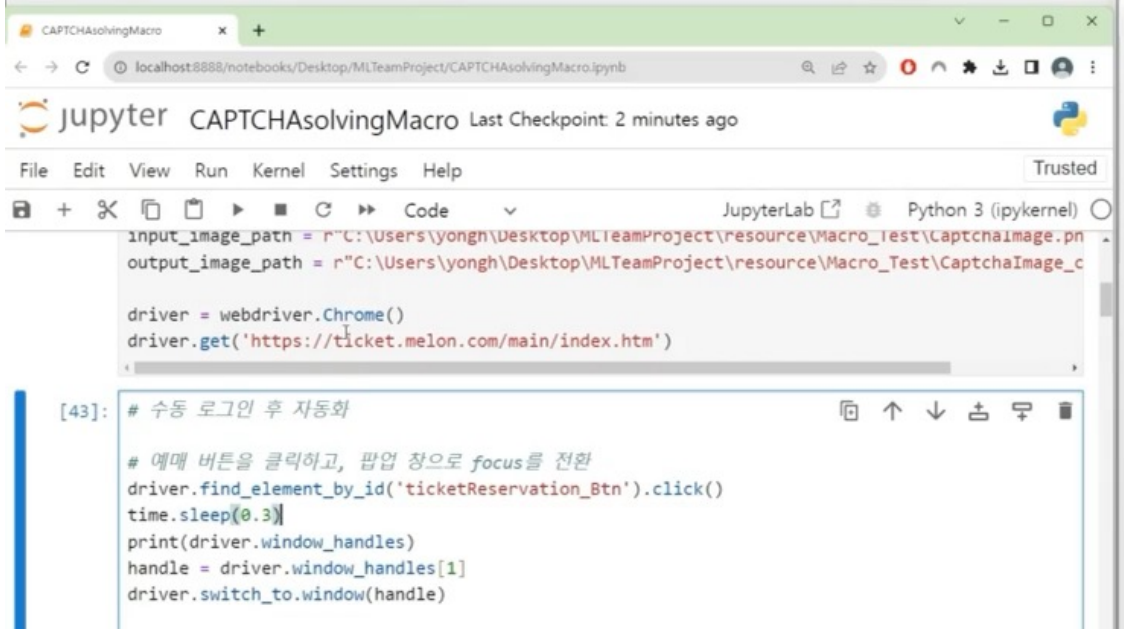
Part 6

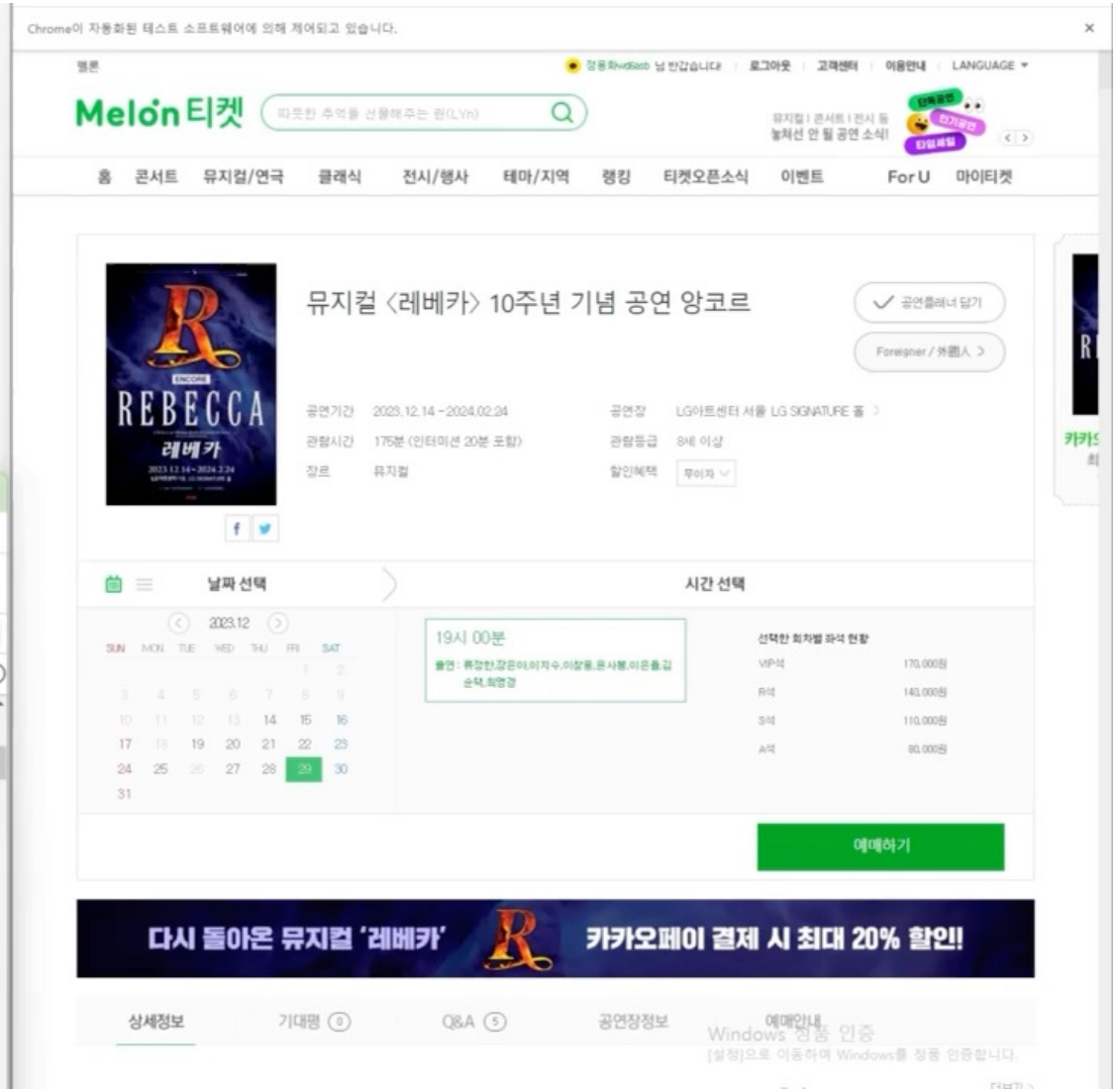
멜론티켓 매크로(Powered by TensorFlow + Selenium Lib)

HUFS 기계학습 7조

<CRNN CAPTCHA Solving MACRO 시연>

수동 로그인 후 [예매하기] 버튼부터 자동!





시연을 위한 동영상입니다.

7

보안 측면에서의 고찰

결론

내가 Captcha를 뚫는 사람이라면...

1. **범용적인 모델을 제작하는 것은 난해하다!**
   폰트, 노이즈의 종류 등 Captcha마다 그 변수가 다양하다.
   결국, **특정 Captcha에 적합한 모델**을 만들어보자

2. **오픈소스 라이브러리를 사용하는지 확인해보자!**
   멜론티켓의 경우 SimpleCaptcha를 사용한다. 이처럼 오픈소스 Captcha를 사용하는 서비스인지 확인해 보는 것도 좋다.

3. **노이즈나 블러를 활용하여 과적합을 줄여보자!**
   모델이 과적합되지 않도록 Noise나 Blur을 주는 것은 정확도 향상에 많은 도움이 된다.

결론

내가 Captcha 매크로를 방어하는 사람이라면...

1. 오픈소스 라이브러리에 의존하지 말자!

2. Captcha 데이터를 다양하게 구성하자!

최대한 많은 변수를 활용하는 것이 방어를 효과적으로 할 수 있다.
대소문자 중 하나만 쓰는 것보단 혼용 하는 것을, 기왕이면 숫자도 추가하자

3. 오답 비중이 높은 문자를 전략적으로 사용하자.

숫자 '0'과 'o', 'C'/'c', 'r'/'v'와 같이 오답이 발생하는 문자를 더 높은
빈도로 등장하게 하는 것이 도움이 될 수 있다.

8

참고문헌

Part 8

결론

1. 이미지 출처
   https://www.pinterest.co.kr/pin/642044490653600077/ 레고 모나리자
   https://ko.wikipedia.org/wiki/%EB%AA%A8%EB%82%98%EB%A6%AC%EC%9E%90 / 모나리자
   https://gall.dcinside.com/mgallery/board/write/?id=aoegame 디씨 캡챠
   https://support.withings.com/hc/en-us/community/posts/4957303667089-Health-report-not-enough-data- / not enough to... 이미지
   https://ticket.melon.com/main/index.htm / 멜론티켓
   https://keras.io/examples/vision/captcha_ocr/ keras crnn
   혼자서 공부하는 딥러닝(한빛출판사, 박해선) / CNN, RNN 설명

2. 논문
   'An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition' (Baoguang Shi)

3. 사용한 주요 라이브러리
   Pillow, TensorFlow(Keras), OpenCv, Selenium, Simple-Captcha(Java)